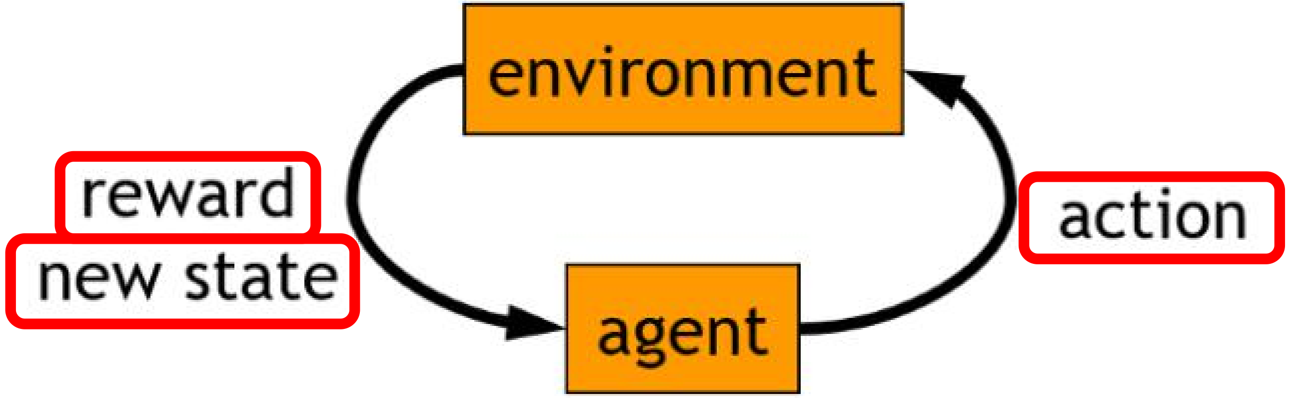
Artificial Intelligence

Q learning

Pham Viet Cuong
Dept. Control Engineering & Automation, FEEE
Ho Chi Minh City University of Technology

Q learning

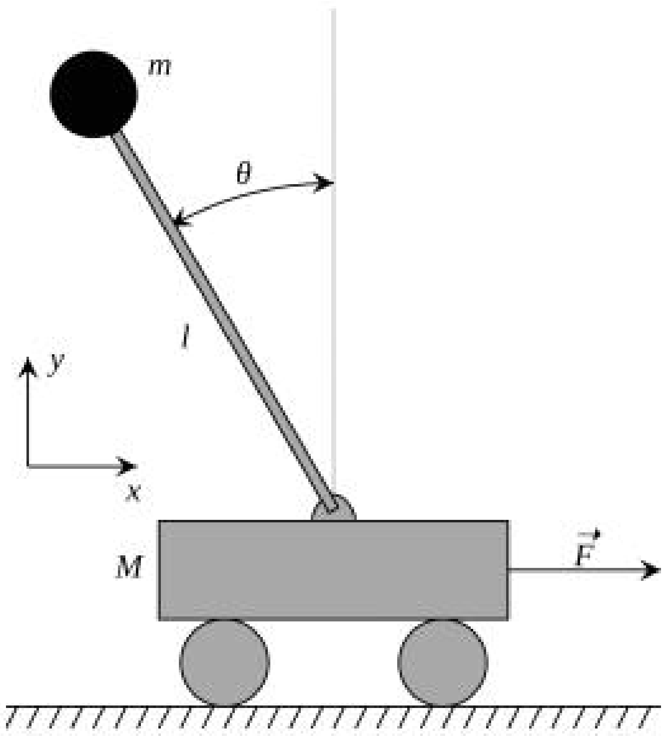
- Supervised learning: Classification, regression
- Unsupervised learning: Clustering
- Reinforcement learning:
  - More general than supervised/unsupervised learning
  - Learn from interactive with environment (perform actions and observe rewards) to achieve a goal
  - Goal: Learn a policy to maximize some measure of long-term reward

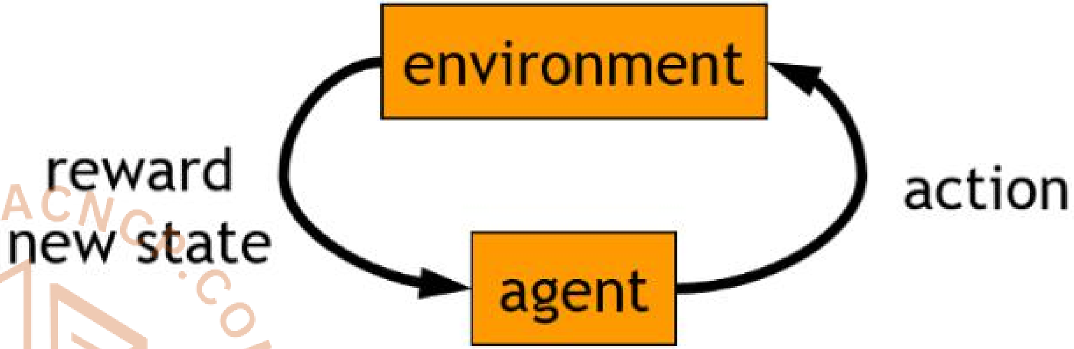


Q learning

✓ Examples:

Cart-Pole Problem





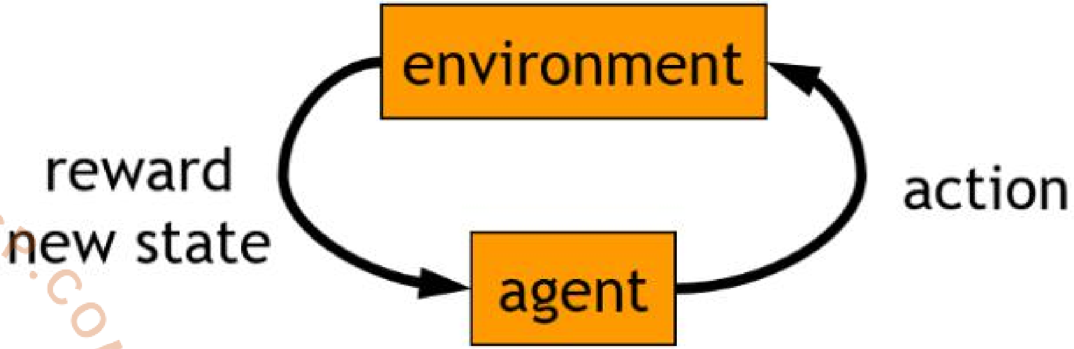
Objective: Balance a pole on top of a movable cart

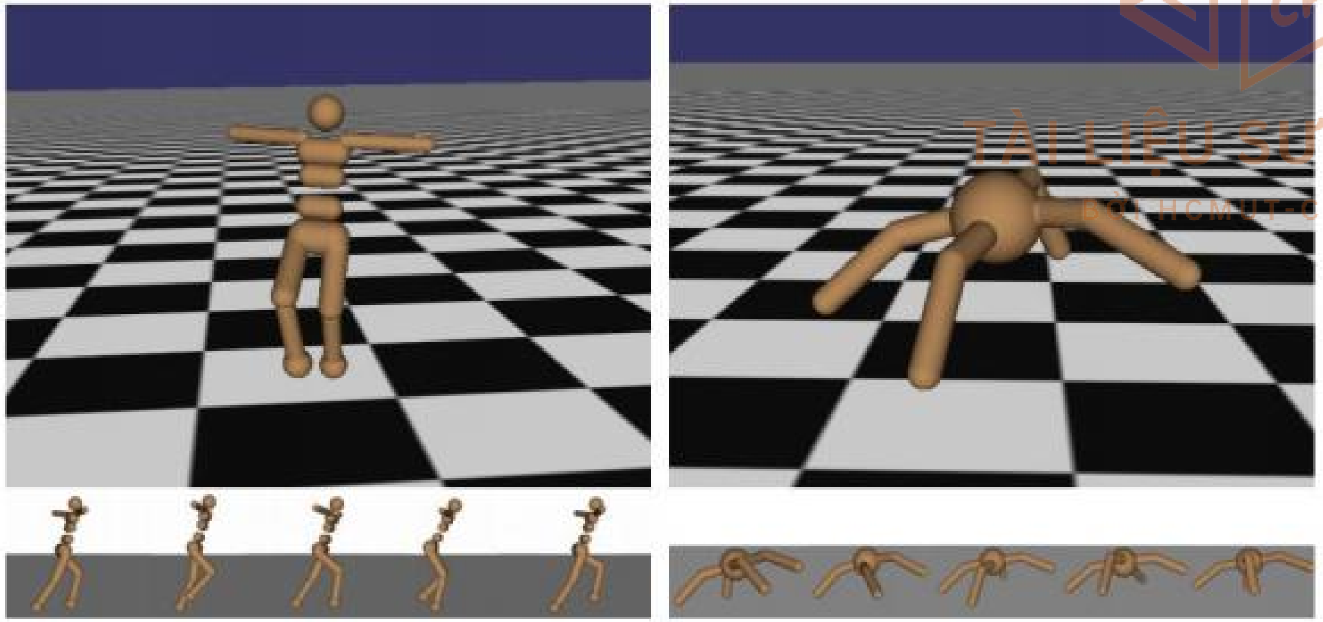
State: angle, angular speed, position, horizontal velocity
Action: horizontal force applied on the cart
Reward: 1 at each time step if the pole is upright

Q learning

✓ Examples:

Robot Locomotion



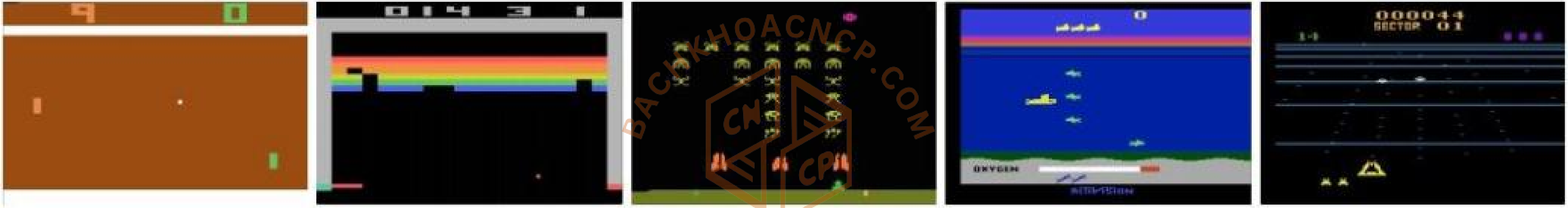


Objective: Make the robot move forward

State: Angle and position of the joints
Action: Torques applied on joints
Reward: 1 at each time step upright + forward movement

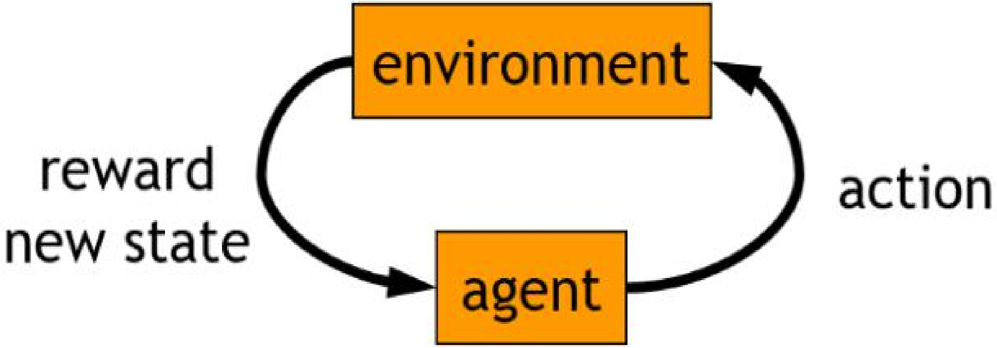
Q learning

✓ Examples: video games



Objective: Complete the game with the highest score

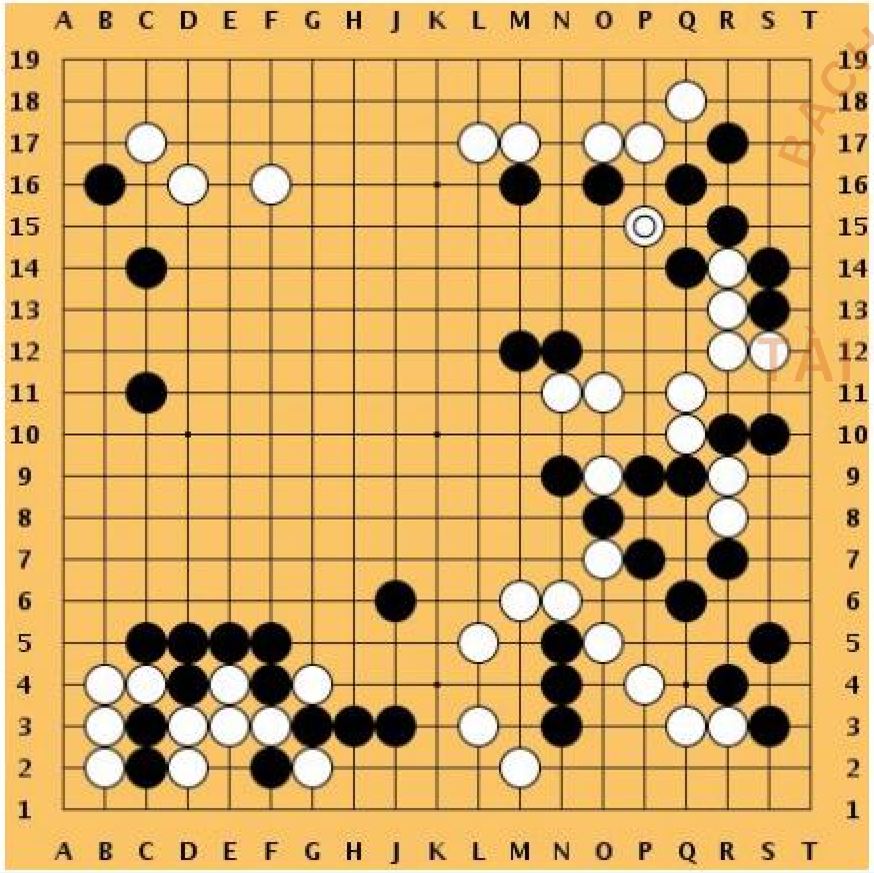
State: Raw pixel inputs of the game state
Action: Game controls e.g. Left, Right, Up, Down
Reward: Score increase/decrease at each time step

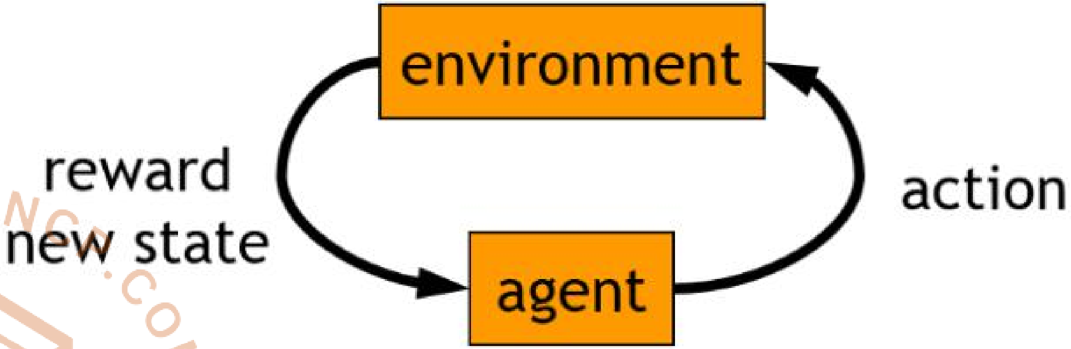


Q learning

✓ Examples:

Go



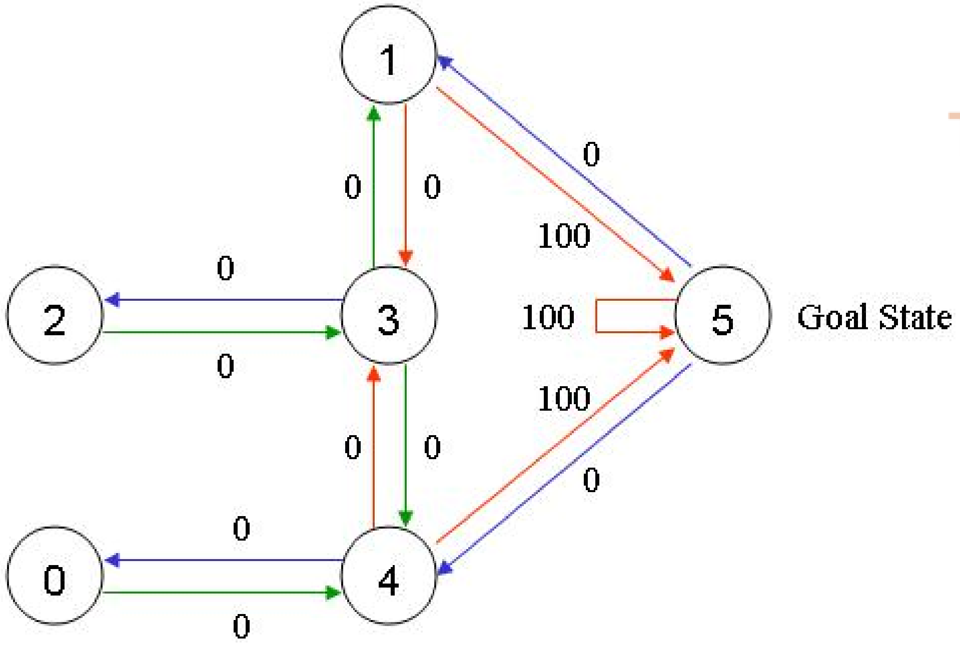


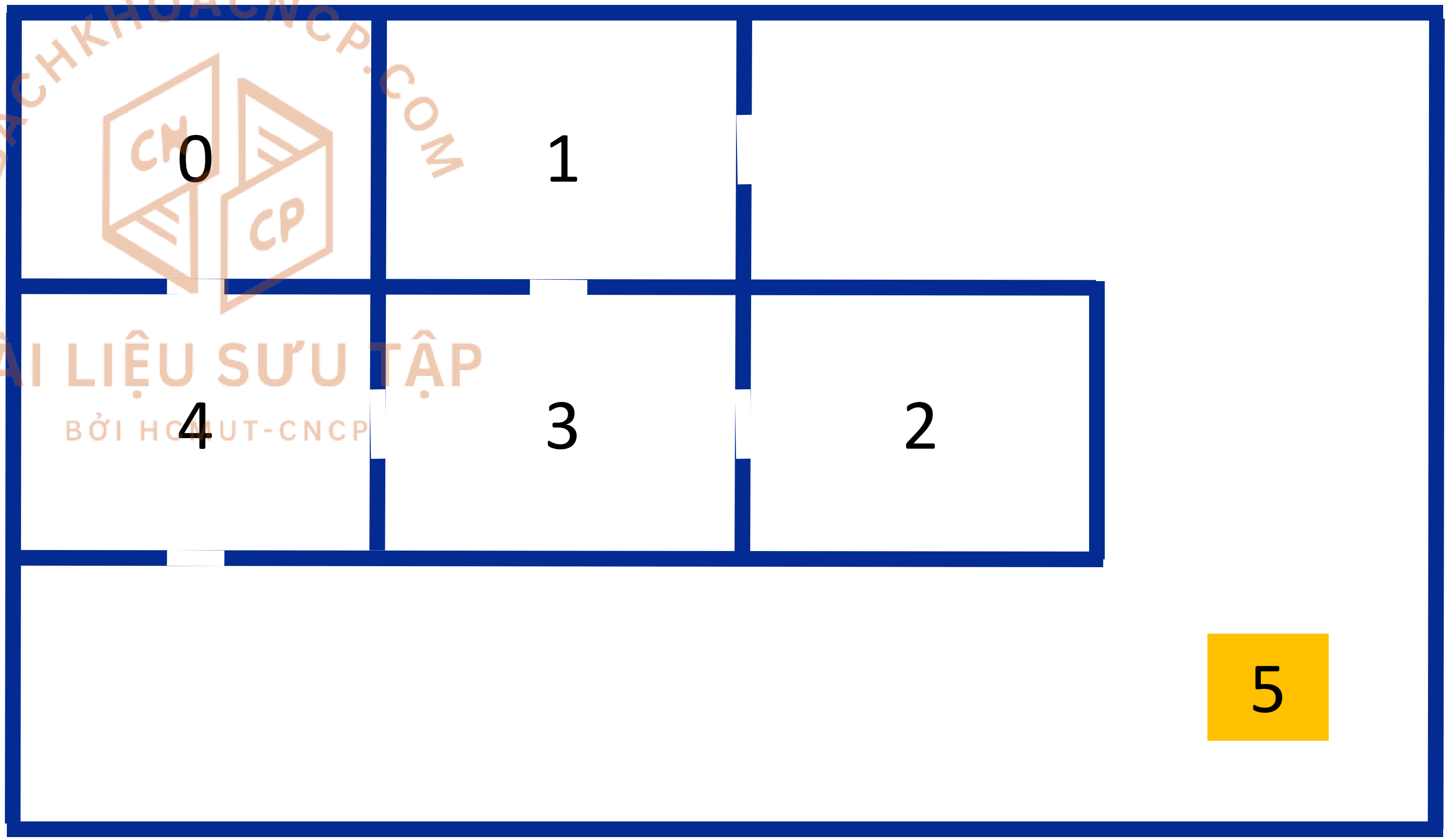
Objective: Win the game!

State: Position of all pieces
Action: Where to put the next piece down
Reward: 1 if win at the end of the game, 0 otherwise

Q learning

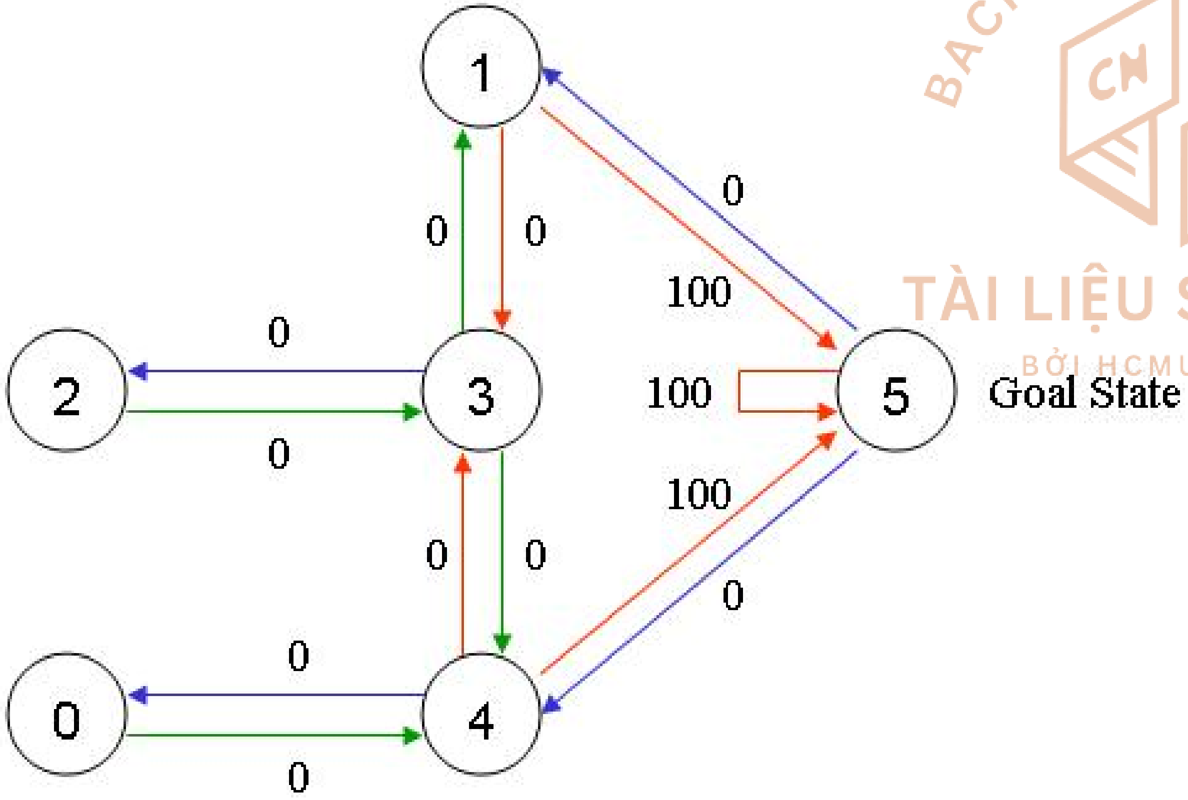
- Example:
  - Put an agent in any room
  - Goal: go to Room 5 with fastest route





Q learning

- State: Room 0, Room 1, . . ., Room 5
- Action: Go to Room 0, Go to Room 1, . . ., Go to Room 5
- Reward: matrix R



Action

$$R = \begin{array}{c|c} \text{State} & \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \left[\begin{array}{cccccc} \; & \; & \; & \; & \; & \; \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{array} \right] \end{array}$$

Q learning

- ✓ Matrix Q: memory of what agent has learned through experience
  - Agent starts out knowing nothing
  - Q is initialized to zero

$$Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Q learning

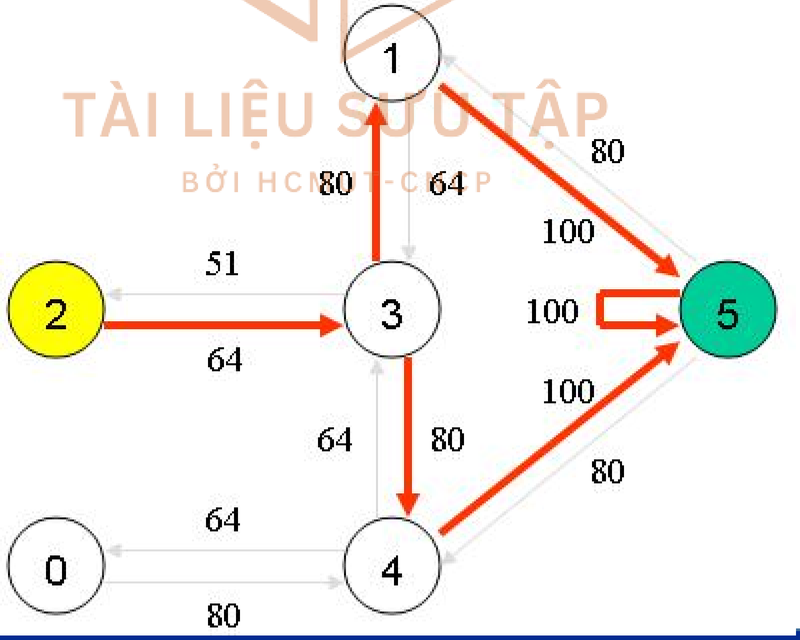
- ✓ Defined:
  - ❖ States
  - ❖ Actions
  - ❖ Rewards matrix R
  - ❖ Matrix Q
- ✓ Training in progress
  - ❖ Updating matrix Q

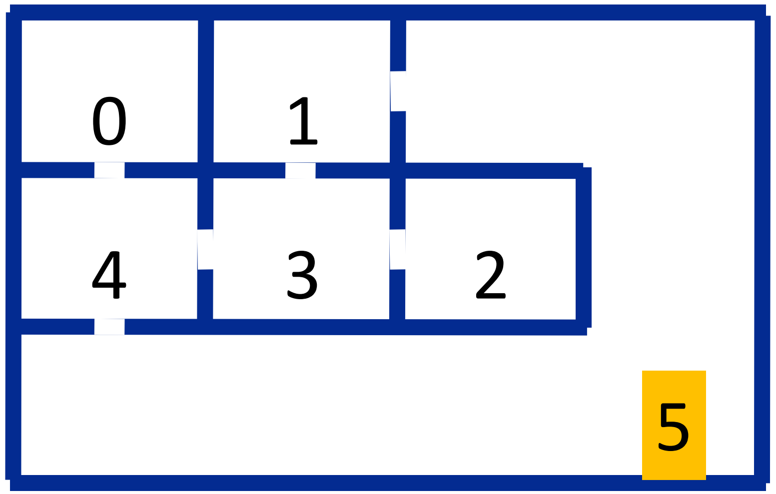
Q learning

✓ Utilize the Q matrix:

- Step 1: Set current state = initial state.
- Step 2: From current state, find the action with the highest Q value.
- Step 3: Perform action chosen in Step 2
- Step 4: Set current state = next state.
- Step 5: Repeat Steps 2, 3 and 4 until current state = goal state.

$$Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 400 & 0 \\ 0 & 0 & 0 & 320 & 0 & 500 \\ 0 & 0 & 0 & 320 & 0 & 0 \\ 0 & 400 & 256 & 0 & 400 & 0 \\ 320 & 0 & 0 & 320 & 0 & 500 \\ 0 & 400 & 0 & 0 & 400 & 500 \end{bmatrix} \end{array}$$





Q learning

✓ Q learning algorithm:

Set the gamma parameter, and environment rewards in matrix R.

Initialize matrix Q to zero.

For each episode:

Select a random initial state.

While the goal state hasn't been reached.

- Select (randomly) one among all possible actions for the current state.
- Using this possible action, consider going to the next state.
- Get maximum Q value for this next state based on all possible actions.
- Compute: $Q(\text{state, action}) \leftarrow \underbrace{R(\text{state, action})}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} * \underbrace{\mathrm{Max}[Q(\text{next state, all actions})]}_{\text{estimate of optimal future value}}$
- Set the next state as the current state.

End While

End For

Q learning

✓ Q learning algorithm: gamma = 0.8, episode 1, initial state: 1

$$
\begin{array}{c|c}
 & \text{Action} \\
\text{State} & \begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array}
\end{array}
$$

$$
R = \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{bmatrix}
-1 & -1 & -1 & -1 & 0 & -1 \\
-1 & -1 & -1 & 0 & -1 & \boxed{100} \\
-1 & -1 & -1 & 0 & -1 & -1 \\
-1 & 0 & 0 & -1 & 0 & -1 \\
0 & -1 & -1 & 0 & -1 & 100 \\
-1 & 0 & -1 & -1 & 0 & 100
\end{bmatrix}
$$

state = 1 action: go to 5 next_state = 5

$$
Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & \boxed{0} & 0 & 0 & \boxed{0} & \boxed{0}
\end{bmatrix}
\end{array}
$$

For each episode:

Select a random initial state.

While the goal state hasn't been reached.

- Select (randomly) one among all possible actions for the current state.
- Using this possible action, consider going to the next state.
- Get maximum Q value for this next state based on all possible actions.
- Compute: Q(state, action) ← R(state, action) + γ*Max[Q(next state, all actions)]
- Set the next state as the current state. 100 0

0.8

End While

End For

$$
Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 100 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{array}
$$

Q learning

✓ Q learning algorithm: episode 2, initial state = 3

state = 3 action: go to 1 next_state = 1

For each episode:

Select a random initial state.

While the goal state hasn't been reached.

- Select (randomly) one among all possible actions for the current state.
- Using this possible action, consider going to the next state.
- Get maximum Q value for this next state based on all possible actions.
- Compute: Q(state, action) ← R(state, action) + γ*Max[Q(next state, all actions)]
- Set the next state as the current state.

$$Q(3,1) \leftarrow \underbrace{0}_{R(3,1)} + \underbrace{0.8}_{\gamma} \cdot \underbrace{100}_{\max Q(1,\cdot)}$$

End While

End For

Action

$$R = \begin{array}{c|c} \text{State} & \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 & -1 & 100 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & \boxed{0} & 0 & -1 & 0 & -1 \\ 0 & -1 & -1 & 0 & -1 & 100 \\ -1 & 0 & -1 & -1 & 0 & 100 \end{bmatrix} \end{array}$$

$$Q = \begin{array}{c|c} & \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \boxed{0} & 0 & \boxed{100} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

$$Q = \begin{array}{c|c} & \begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 \end{matrix} \\ \begin{matrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 100 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 80 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Q learning

✓ Q learning algorithm: episode 2, initial state = 3

state = 1   action: go to 5

For each episode:

Select a random initial state.

While the goal state hasn't been reached.

- Select (randomly) one among all possible actions for the current state.
- Using this possible action, consider going to the next state.
- Get maximum Q value for this next state based on all possible actions.
- Compute: Q(state, action) ← R(state, action) + γ*Max[Q(next state, all actions)]
- Set the next state as the current state.

100   0

0.8

End While

End For

$$
Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 400 & 0 \\
0 & 0 & 0 & 320 & 0 & 500 \\
0 & 0 & 0 & 320 & 0 & 0 \\
0 & 400 & 256 & 0 & 400 & 0 \\
320 & 0 & 0 & 320 & 0 & 500 \\
0 & 400 & 0 & 0 & 400 & 500
\end{bmatrix}
\end{array}
$$

Action

$$
R = \begin{array}{c} \text{State} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
-1 & -1 & -1 & -1 & 0 & -1 \\
-1 & -1 & -1 & 0 & -1 & \boxed{100} \\
-1 & -1 & -1 & 0 & -1 & -1 \\
-1 & 0 & 0 & -1 & 0 & -1 \\
0 & -1 & -1 & 0 & -1 & 100 \\
-1 & 0 & -1 & -1 & 0 & 100
\end{bmatrix}
\end{array}
$$

$$
Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 100 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 80 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & \boxed{0} & 0 & 0 & \boxed{0} & \boxed{0}
\end{bmatrix}
\end{array}
$$

$$
Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \end{array} \\
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 100 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 80 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{array}
$$

Q learning

✓ Q learning algorithm:

Set the gamma parameter, and environment rewards in matrix R.

Initialize matrix Q to zero.

For each episode:

Select a random initial state.

While the goal state hasn't been reached.

- Select (randomly) one among all possible actions for the current state.
- Using this possible action, consider going to the next state.
- Get maximum Q value for this next state based on all possible actions.
- Compute: $Q(\text{state, action}) \leftarrow \underbrace{R(\text{state, action})}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} * \underbrace{Max[Q(\text{next state, all actions})]}_{\text{estimate of optimal future value}}$
- Set the next state as the current state.

End While

End For

Q learning

✓ Q learning algorithm:

Set the gamma parameter, and environment rewards in matrix R.

Initialize matrix Q to zero.

For each episode:

Select a random initial state.

While the goal state hasn't been reached.

- Select (randomly) one among all possible actions for the current state.
- Using this possible action, consider going to the next state.
- Get maximum Q value for this next state based on all possible actions.
- Compute: Q(state, action) ← $\boxed{(1-\alpha)\text{*}\underbrace{Q(\text{state, action})}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}}}$ * {$\underbrace{R(\text{state, action})}_{\text{reward}}$ + $\underbrace{\gamma}_{\text{discount factor}}$*$\underbrace{\text{Max}[Q(\text{next state, all actions})]}_{\text{estimate of optimal future value}}$}
- Set the next state as the current state.

End While

End For

old value learning rate reward

discount factor estimate of optimal future value

Artificial Neural Networks

- References
  - http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture14.pdf
  - http://mnemstudio.org/path-finding-q-learning-tutorial.htm